# Landsat 7
# Processing System (LPS)
# Software Management Plan

**June 1, 1995**

**GODDARD SPACE FLIGHT CENTER**
**GREENBELT, MARYLAND**

## Abstract

The Landsat 7 Processing System (LPS) provides Landsat 7 data receipt and processing support to the Landsat 7 program, in conjunction with the Earth Science Mission Operations (ESMO) Project. The LPS receives raw wideband data from the Landsat 7 Ground Station, located at the Earth Resources Observation System (EROS) Data Center (EDC), processes it into Level 0R, browse and metadata files, and provides them to the Land Processes Distributed Active Archive Center (LP DAAC), also located at the EDC. The Software Management Plan presented in this document outlines a strategy for the management of the LPS software design, development, integration, test, and maintenance.

**Keywords:**

Landsat 7
Landsat 7 Processing System (LPS)
Landsat 7 Ground Station (LGS)
Land Processes Distributed Active Archive Center (LP DAAC)
Earth Resources Observation System (EROS)
EROS Data Center (EDC)
Functional and Performance Specification (F&PS)
Mission Operations and Data Systems Directorate (MO&DSD)
Mission Operations and Systems Development Division (MOSDD)

# Preface

This document outlines a strategy for the management of the Landsat 7 Processing System (LPS) software design, development, integration, test, and maintenance. This document, once baselined will be controlled by the Mission Operations and Systems Development Division (MOSDD) Code 510 configuration control board (CCB) and maintained and updated, as required, by the LPS Project.

This document was prepared by:

Landsat 7 Processing System Project
Code 510
Goddard Space Flight Center
Greenbelt, MD 20771

GSFC
M. Dowdy
T. Grubb
J. Henegar
J. Hosler
E. Lee
R. McIntosh
K. Michael
J. Rosenberg
D. Sames
R. Schweiss
G. Toth

SEAS
D. Alban
T. Aslam
B. Boyce
D. Crehan
J. Freeman
S. Gottsagen
A. Hall
M. Huang
H. Kim
J. Lauderdale
G. Lakshmi
C. Liu
S. Liu
N. Patel
S. Priest
R. Shea
D. Specht
R. Tassi
S. Tseng
R. Tingley
M. Wong

# Table of Contents

# 1 Introduction

## 1.1 Purpose and Scope

The purpose of this document is to outline the strategy for the management of the Landsat 7 Processing System (LPS) software design, development, integration, test, and maintenance.

The software will be developed by Goddard Space Flight Center (GSFC) and Systems, Engineering, and Analysis Support Services (SEAS) personnel.

## 1.2 Document Organization

This document is divided into six sections. Section 1 provides a brief overview of the LPS and its functionality. Section 2 describes the various responsibilities of the organizations involved in the development and continued maintenance of the LPS. Section 3 describes the responsibilities of the civil servants necessary for the successful development of the LPS software. Section 4 describes in detail the software methodology to be used in the development and testing of the LPS, the approach to be used for configuration management, product assurance, and training. In addition, this section outlines the metrics that will be kept during the development life-cycle and the guidelines to be followed during the design and implementation of the system. Section 5 describes the manner in which the SEAS task for the development of the LPS software will be evaluated, and the final section discusses the responsibilities of the software development group with regards to the installation and transition of the system to operations and for continuing maintenance.

## 1.3 Document Maintenance

This document will be included as an appendix to the Project Management Plan (PMP) and maintained by the LPS software manager. The Software Management Plan will be updated as necessary to ensure the successful software development of the LPS.

## 1.4 Project Overview

The LPS is being developed by the Mission Operations and Systems Development Division (MOSDD) Code 510 of the Mission Operations and Data Systems Directorate (MO&DSD) at the Goddard Space Flight Center (GSFC). The system will be delivered to the Earth Resources Observation System (EROS) Data Center (EDC) in Sioux Falls, South Dakota. The National Oceanic and Atmospheric Administration (NOAA) will be responsible for the continuing operations and maintenance of the system.

The LPS provides processing for the wideband data generated by the Enhanced Thematic Mapper Plus (ETM+) instrument onboard the Landsat 7 spacecraft. The LPS performs Level 0R processing on the data received from the Landsat 7 Ground Station (LGS). In addition, metadata and browse products are generated. All of these products are made available to the Land Processes Distributed Active Archive Center (LP DAAC) for distribution to the end users.

The LGS forwards four physical channels of data at 75Mbps each during a single contact with the Landsat 7 spacecraft to the LPS for data capture. After a contact period is captured, the data is then Level 0R processed at a lower rate. For each stream of data, there is a separate functionally independent string of processing functions in the LPS. These functional processing strings are identical in terms of hardware and software configuration. The LPS will provide processing for the equivalent of 250 scenes per day.

For a more detailed overview of the LPS, consult the LPS Operations Concept document.

# 2 Software Development Responsibilities

The LPS software will be jointly development by GSFC and SEAS personnel. As the subsystems and responsibilities are defined and refined during each development life-cycle phase, the following lists will be updated to reflect the distribution of the responsibilities to the appropriate design teams.

## 2.1 GSFC

Specific portions of the LPS will be developed by an in-house development team. The functions to be implemented by the in-house team are listed by phase:

Software Requirements Specification

- Each developer will be assigned to a particular LPS subsystem.
- The developer will participate with SEAS counterpart(s) in specifying the functional requirements for the assigned subsystem:
    - Data Flow Diagrams
    - Data Dictionary
    - Functional/Process specifications
    - Requirement Traceability
    - Software Requirement Specification (SRS) text descriptions
- Each developer will participate in software trade studies as time permits.

Preliminary Design

- Each developer will be assigned to a particular LPS subsystem.
- Each developer will participate with SEAS counterpart(s) in defining the interface design definitions for the assigned subsystem.
- Each developer will participate with SEAS counterpart(s) in specifying the design for the assigned subsystem:
    - Preliminary Structure Charts
    - Refined Data Dictionary
    - Module Specifications
        - Relaxed Primitive Design Language (PDL) for main unit
        - Descriptions for all other units
    - Preliminary Design text descriptions for inclusion into the Preliminary Design Specification (PDS) document
- Each developer will participate in software trade studies and prototypes as time and need permit.

Critical Design

- Each developer will be assigned to a particular LPS subsystem.
- Each developer will participate with SEAS counterpart(s) in refining and completing the interface design definitions for the assigned subsystem.

- Each developer will participate with SEAS counterpart(s) in completing the design for the assigned subsystem:
    - Final Structure Charts
    - Final Data Dictionary
    - Module Specifications with relaxed PDL for all units
    - Detailed Design text descriptions for inclusion into the final Detailed Design Specification (DDS) document
- Each developer will participate in software trade studies and prototypes as time and need permit.

Implementation

- Each developer will be assigned specific Software Configuration Items (SWCIs).
- Each developer will be responsible for the implementation of their assigned SWCIs.
    - Unit PDL
    - Coding
    - Unit Testing
    - Participation in System Integration Testing
- Each developer will be responsible for the requirement tracking to the Software Requirement Specification (SRS) of their assigned SWCIs.

## 2.2 SEAS

The majority of the LPS will be developed by a SEAS development team. The functions to be implemented by the SEAS development team are listed by phase:

Software Requirements Specification

- Each developer will be assigned to a particular LPS subsystem.
- The developer will participate with GSFC counterpart(s) in specifying the functional requirements for the assigned subsystem:
    - Data Flow Diagrams
    - Data Dictionary
    - Functional/Process specifications
    - Database Design Specification
    - Requirement Traceability
    - Software Requirement Specification (SRS) text descriptions
- Each developer will participate in software trade studies as time permits.

Preliminary Design

- Each developer will be assigned to a particular LPS subsystem.
- Each developer will participate with GSFC counterpart(s) in defining the interface design definitions for the assigned subsystem.

- Each developer will participate with GSFC counterpart(s) in specifying the design for the assigned subsystem:
    - Preliminary Structure Charts
    - Refined Data Dictionary
    - Module Specifications
        - Relaxed PDL for main unit
        - Descriptions for all other units
    - Logical Database Design
    - Preliminary Design text descriptions for inclusion into the Preliminary Design Specification (PDS) document
- Each developer will participate in software trade studies and prototypes as time and need permit.

Critical Design

- Each developer will be assigned to a particular LPS subsystem.
- Each developer will participate with GSFC counterpart(s) in refining and completing the interface design definitions for the assigned subsystem.
- Each developer will participate with GSFC counterpart(s) in completing the design for the assigned subsystem:
    - Final Structure Charts
    - Final Data Dictionary
    - Module Specifications with relaxed PDL for all units
    - Physical Database Design
    - Detailed Design text descriptions for inclusion into the final Detailed Design Specification (DDS) document

- Each developer will participate in software trade studies and prototypes as time and need permit.

Implementation

- Each developer will be assigned specific Software Configuration Items (SWCI).
- Each developer will be responsible for the implementation of their assigned SWCIs.
    - Unit PDL
    - Coding
    - Testing as described in section 4.1.7
- Each developer will be responsible for the requirement tracking to the Software Requirement Specification (SRS) of their assigned SWCIs.

In addition to the development responsibilities, SEAS is responsible for the system administration of the LPS development and test platforms. This includes, but is not limited to the following:

- Installation of COTS software
- Maintaining the hardware/software inventory reports
- System administration/backups
- Resolution of LPS network problems

## 2.3 Other Organizations

The LPS will be utilizing the Generic Telemetry Simulator (GTSIM) for test data generation. Modifications to incorporate the data formats for the Landsat 7 mission into GTSIM are currently planned or underway. The modifications are being performed by GSFC and Network and Mission Operations Support (NMOS) personnel.

It is proposed that Acceptance Test activities will be conducted by EDC personnel. This is to be negotiated with EDC. A Factory Acceptance Test (FAT) will be conducted at GSFC, with EDC personnel in attendance, before the installation of the LPS system at the EDC. The responsibility for the organization and planning of this FAT will also be negotiated with EDC. In addition, the continuing maintenance of the LPS software will transition to EDC personnel at an agreed upon time.

# 3 GSFC Software Management Organization

This section describes the roles and responsibilities for the management of the joint GSFC and SEAS software development. As noted previously, the functional software areas of responsibilities will be updated as the subsystems and responsibilities are defined and refined during each development life-cycle phase.

## 3.1 LPS Software Manager

The LPS software manager is responsible for the design, development, and test of all LPS software.

The LPS software manager leads and directs the software requirements, design, development, and test process for the LPS. System integration and testing falls under the responsibilities of the LPS system engineer; the LPS software manager will manage these efforts in conjunction with the LPS system engineer in order to provide continual end-to-end ownership of the software.

The duties of the LPS software manager are listed below:

- Manage budget/schedule for all LPS software development coordinating with the project manager.
- Develop procurement material for the purchase of all equipment necessary for software development (workstations, CASE tools, etc.). This will be done through the Science and Engineering Workstation Procurement (SEWP) system.
- Develop software development project life-cycle to be followed for development. Establish software development quality assurance "standards" to be utilized during development.
- Establish set of metrics to be maintained describing software development progress.
- Serve as official Assistant Technical Representative (ATR) for all SEAS software development efforts.

- Provide overall technical direction to SEAS contractor for operational software development, with assistance from LPS software engineer and LPS in-house functions manager as required.

- Conduct regular status meetings with appropriate SEAS/GSFC personnel as needed.

- Lead efforts to identify/select/procure candidate Commercial Off-the-Shelf (COTS) software packages to be utilized for satisfying LPS data processing requirements.

- Serve as technical lead for the following LPS functions, performing technical duties as necessary for proper management of function development. (Duties described under LPS software engineer). These functions are:

    - Raw Data Capture Subsystem (RDCS)

    - Raw Data Processing Subsystem (RDPS)

    - Major Frame Processing Subsystem (MFPS)

    - Payload Correction Data Processing Subsystem (PCDS)

    - Image Data Processing Subsystem (IDPS)

- Manage in conjunction with the LPS system engineer, the system testing of the LPS. Chair the Configuration Review Board (CRB) to disposition Internal Configuration Change Requests (ICCRs) resulting from all test activities.

- Assist appropriate staff at EDC in Sioux Falls, SD, with the installation, checkout, initial operations, and anomaly resolution for the LPS software after completion of development.

- Define and direct all prototyping efforts for LPS software development with assistance from the LPS software engineer and LPS in-house functions manager as required.

- Interface/coordinate activities with the LPS project manager, including attending external meetings, project documentation review, presenting at Project Status Reviews (PSRs) or other status reviews, answering project action items, etc.

- Interface/coordinate with the LPS system engineer to ensure proper coordination of interfaces, implementation plans, and Interface Control Documentation.

- Interface/coordinate with the LPS hardware manager to ensure proper coordination between custom hardware and software needed to support the custom hardware.

## 3.2 LPS Software Engineer

The LPS software engineer is responsible for the remaining functionality being developed by the SEAS contractor. In addition, the LPS software engineer will act as a central point of contact for the various technical leads to aid the software manager in insuring that there is consistency and reusability throughout the software design and development.

- Lead effort to define the LPS functional subsystem interface definitions. Facilitate the development and creation of the Interface Definitions Document (IDD).

- Provide assistance to LPS software manager by serving as technical lead for the following LPS functions (total not to exceed 14,000 Delivered Source Instructions (DSI)):

    - LPS Data Transfer Subsystem (LDTS)

    - Management and Control Subsystem (MACS)

    - Database

    - Operations Interface

- Perform the following duties as technical lead for the identified functions (development work to be performed by SEAS contract/GSFC personnel):
    - Lead effort to define the software requirements for each function.
    - Lead effort to develop preliminary/critical designs for each function. Lead effort to develop all function-level interface definitions and test plans.
    - Manage code and unit test effort by attending all PDL and code walkthroughs for each function as required.
    - Manage test effort specific to each function. Provide technical analysis, prioritization, and disposition to CCRs/ICCRs related to the above functions.
    - Provide inputs to LPS software manager regarding development progress of the above functions, and SEAS contractor performance.
    - Ensure that above functional area requirements, operational needs, and Interface Control Documentation (ICD) requirements are met.
    - Ensure that each function of the system adheres to all development methodology and project standards.
- Work with the LPS software manager and the LPS system engineer to organize and oversee the system integration and testing as outlined in the System Integration and Test Plan.
- Assist LPS software manager as required for other assignments, such as documentation reviews, attendance at external meetings, preparing/presenting material at status reviews, answering project action items, etc.

- Conduct "task lead" meetings as necessary with representatives from all functions to ensure consistency and reuse of design and implementation.

## 3.3 LPS In-House Functions Manager

A government person will serve as the technical lead for the software to be developed in-house if a functional subsystem, or set of functional subsystems, are designed and/or implemented solely by in-house personnel. The responsibilities for the LPS in-house functions manager are described below. Again, as the system design is finalized, the items selected for in-house development may be modified. The responsibilities of this person will then be modified to reflect the functionality selected for in-house development.

- Define the processing requirements for the TBD function(s) as part of the Software Requirements Specification process.

- Lead in-house development team to develop preliminary and detailed design, code and unit test, and module test software for the TBD function(s) of LPS.

- Perform duties as described under third bullet for LPS software engineer with regards to the functions assigned to the in-house development team.

# 4 Software Development

## 4.1 Methodology

The LPS project will follow the standard waterfall method of software development with input from the SEAS System Development Methodology (SSDM). However, this methodology will be customized to meet the needs of the project.

### 4.1.1 System Requirements/Operations Concept Definition

System Requirements definition falls under the system engineering task; however, the software development team is responsible for understanding, and conforming to these requirements. Members of the software development team will provide comments and insight into the requirements and operations concept definition. It is everyone's responsibility to be aware of and understand the system requirements. The LPS system requirements will be traced back to the Landsat 7 Detailed Mission Requirements by the system engineering group. System engineering will maintain the traceability of requirements through the life of the project.

Prototyping efforts are currently underway to help understand the LPS functionality and identify requirements.

During this phase the LPS Functional and Performance Requirement Specification (F&PS) and Operations Concept documents will be baselined. At the culmination of this phase, a presentation will be given to the MOSDD Configuration Control Board (CCB) for approval.

### 4.1.2 System Design

System design is again the responsibility of the system engineering group. The software development group is responsible for understanding and providing insight to the system design. System design involves breaking the system into the functional subsystems, allocating requirements to those functions, and defining the high level interfaces between these functions. This includes identifying requirements to be

met by software, hardware, etc. and determining manual and automated functions. The requirements to be met by the software are allocated to individual SWCIs, correspondingly, Hardware Configuration Items (HWCIs) are identified.

In addition, this phase will also provide the system architecture. Prototyping, rapid development, bench marking, modeling, etc. will continue during this phase to select and validate the system design and architecture.

In summary, the activities to be performed during this phase are: selection of the system architecture, generation of the system design including system level data flow diagrams and data dictionary, allocation of requirements to functional subsystems, traceability of requirements to and from the system requirements, and further refinement of the operational scenarios. During this period as well, the generation of external Interface Control Documents (ICDs) will be initiated.

Deliverables for this phase include:

- System Design Specification (System Engineering [SE])
    - Data Dictionary
    - System Level Data Flow Diagrams
    - Requirement Traceability
    - Refined Operational Scenarios
    - Prototyping results
    - System Architecture
- Preliminary External ICDs (SE)
- Informal System Design Review (SE)

NOTE: After each deliverable, groups with the primary responsibility for the deliverable are identified in parenthesis. However each deliverable will require coordination between the LPS system engineering and the development groups for the project to succeed.

### 4.1.3 Software Requirements

This phase will provide analysis of the requirements assigned to each functional subsystem. Data flow diagrams and functional/process specifications will be generated for the software requirements analysis, and for the hardware requirements analysis where appropriate.

At this phase, the responsibility of the IDD will be transferred to the software development group from the system engineering group. In the past this document has remained the responsibility of the system engineering group. However, since the development group is the stake holder in this document, the responsibility for the further generation and maintenance of this document will be accomplished by the software development group. System's engineering **remains** responsible for the external interfaces. In addition system engineering will remain responsible for the requirements traceability to and from the system requirements.

Other activities during this phase will be the identification of potential COTS products to be used in the development or for operational use.

As always, prototyping will continue for risk mitigation of technical issues areas and to determine feasibility of requirements as necessary.

Deliverables for the phase include:

- Subsystem Software/Hardware Requirements Specification
    - Data Dictionary (Software Development [SD])
    - Software Requirement Data Flow Diagrams (SD)
    - Hardware Requirement Data Flow Diagrams (SD/Hardware Development [HD])

- Functional/Process Specifications (SD)
- Operational Scenarios (SE)
- Requirements Traceability (SE/SD)

- Formal CCB review of the System Design and Software Requirements Specification (SE/SD)

### 4.1.4 Preliminary Design

The preliminary design phase transforms the software requirements to the software architecture. The preliminary design provides high level structure charts for the software. A high level Build Implementation Plan will be generated and mapped to the software requirements. The Build Implementation Plan is the responsibility of the system engineering group, but will require significant inputs from the software development group. Final versions of the ICDs will be generated so the detailed design may be completed. Software development will need to review and understand these ICDs.

During this phase, additional COTS products to be used in the development or operational system will be identified and procured.

Also, during this phase, all test tool and test environment needs by the developers or test groups will be identified for design and implementation.

Again, prototyping will be conducted for key risk areas and to resolve technical issues. Prototyping of the operations interface will begin at this time for early feedback from the operations group.

Deliverables for this phase include:

- Preliminary Design Specification (SD)
  - Preliminary Database Design Specification/Logical Schema (SD)
  - Preliminary Operations Interface Specification (SD)
  - Preliminary Structure Charts (SD)

- Module Specifications (SD)
    - Relaxed PDL for the main unit
    - Descriptions for all other units
- Refined Data Dictionary (SD)
- Refined Operations Scenarios (SE)
- Requirement Traceability (SE)

- Preliminary System Integration and Test Plan (SE)
- Final ICDs (SE)
- Preliminary Build Implementation Plan (SE)

### 4.1.5 Detailed Design

The detailed design phase will completely decompose the preliminary design into detailed structure charts for each unit. Generation of prologs and English like PDL to describe the functions of each unit will be completed. More detailed prototyping of the operations interface will continue to provide further input from the operations group. Any other technical issues or risk areas may be prototyped as well.

Deliverables for this phase include:

- Detailed Design Specification:
    - Final Database Design Specification/Physical Schema (SD)
    - Detailed Operations Interface Specification (SD)
    - Final Structure Charts (SD)
    - Final Module Specifications including relaxed PDL for all units (SD)
    - Final Data Dictionary (SD)

- Final Operations Scenarios (SE)
- Requirement Traceability (SE)
- Preliminary Operations Guide (SD)

- Final System Integration and Test Plan (SE)
- Final Build Plan (SE)

### 4.1.6 Implementation

The implementation phase is typically broken into three stages, the unit design which results in formal PDL, coding of the unit, and conducting unit and module testing. Walkthroughs will be held for each unit for each of these phases as outlined in section 4.5.1. The functionality to be implemented for each build and release is documented in the LPS Build Implementation Plan.

Detailed Design Specification documentation and Operations Guide will be kept up to date for each release.

Deliverables for this phase include:

- Final Operations Guide (SD)
- Source Code for each Release (SD)
- Detailed Design Specification updated to the as-built documentation (SD)
- Unit Test Plans and Execution (SD)
- Module Test Plans and Execution (SD)
- Formal PDL and Code Walkthroughs (SD)
- Informal BRR and BDRs for each build other than the first (SD)
- Detailed System Test Procedures (SE)

Prior to the start of coding for each build (other than the first build), the LPS will conduct two internal reviews.

#### 4.1.6.1 Build Readiness Review (BRR)

The first review prior to the start of coding for each build will be the Build Readiness Review (BRR) which will be conducted within two weeks of the beginning of work on a build. The purpose of the BRR is to provide an overview of the build requirements, updates on prototyping activities, and identification of open issues related to the build. Preparation for this review is small: retrieving and identifying changes from the plans presented at the preliminary and detailed design phases. There will be a single BRR which will include inputs from each software subsystem.

#### 4.1.6.2 Build Design Review (BDR)

Once all of the PDL for a given build is completed, the second review called the Build Design Review (BDR) will be conducted. The BDR will review in detail the data flow and structure of the software architecture. PDL for critical units as identified by the project will be presented. Resolutions from the BRR will be presented.

There will be a staggered BDRs, one for each software subsystem. This will allow subsystems that complete the assigned PDL early to begin immediately on the coding of the reviewed units. In the event that there are multiple subsystems that complete the assigned PDL at the same time, a single BDR encompassing the associated subsystems may be feasible.

#### 4.1.6.3 Coding

Coding will commence after the BDRs. Code will follow the standards put forth by the project. Developers will be responsible for performing unit and module testing for the units which they develop. Section 4.2 covers these testing areas in more detail.

### 4.1.7 Testing

The following provides a summary of the phases of testing to be completed for the LPS software. Test plans, procedures, and summaries will be documented for each of the phases.

Whenever possible the identical configuration for both the hardware and software should be used for the development, test, and operational environment. This will provide two advantages: problems discovered in system testing or operations will be repeatable in the development environment, and a smoother integration with the hardware at the beginning of each promotion to a different environment.

#### 4.1.7.1 Unit and Module

Unit and module testing are to be performed by the developers who code the associated units or modules Unit testing involves testing each individual component to ensure that they operate as expected. Unit testing treats each component as a stand-alone entity which does not need other components during the testing phase. A module is a collection of dependent components which are tested together independently of other modules within the system.

A test coverage tool will be used to aid the developer in documenting that each and every path through the software has been touched. This will allow the developer to concentrate on the tests and inputs which will be needed to accomplish this task. Informal test procedures will be documented during this phase. The results of the test coverage tool will be used to verify execution of the procedures.

#### 4.1.7.2 Build Integration

Build integration testing will take place as the last step of the implementation for the build. This will be performed by members of the development teams. The objective is to validate the interfaces and functionality of the individual subsystems, with regard to the Build Implementation Plan, by integrating the software modules which comprise the build.

At the completion of this phase of testing, a single software entity will be provided to system integration and test for integration of the entire system. This is synonymous with what the Packet Processor II Data Capture Facility (Pacor II) is calling the development integration and test. Given the small size of each LPS subsystem, however, we will expand this phase to include all software implemented in a build which will include government developed software.

#### 4.1.7.3 System Integration

Integration and testing of the system is the phase in which the software and hardware are integrated together in the test environment. This phase will concentrate on validating the interfaces between the functional subsystems.

At the completion of this phase of testing, a single entity is ready for system verification and validation.

This phase will be organized and completed by the LPS system engineering group with support from the LPS software development group in problem identification and resolution as necessary.

#### 4.1.7.4 System Testing

System testing treats the system as a black box for testing, and validates and verifies the system based on the Build Implementation Plan. Test status will be provided as directed by project personnel.

The LPS is considering a new approach in this area. Typically on a project, separate independent system test personnel provide the system test plan, procedures, and execution of the tests.

The LPS would like to look at using people already assigned to the project and know the system, such as the LPS system engineers, to prepare the system test plan. Minimal work necessary should be performed to ensure that we are ready for testing after the first build. The system test plan will include identifying test tools, identifying test environment needs, and very high level test planning. The scheduled duration of the first build implementation will provide ample time to prepare a detailed test plan and procedures. Of course, the draw back of this approach is that we lose some of the "independent"

testing. The approach does, however, provide a smoother staffing curve for the LPS system engineering task and provides advantages in the long run by using people experienced with the system to test the system. This approach to our testing is still TBR.

Other system test functions will be: to provide demonstrations of the LPS to the project and MOSDD management personnel, and to provide training to the acceptance test personnel. This is further outlined in section 4.8.

#### 4.1.7.5 Acceptance Testing

This is the final stage in the testing process before the system is accepted for operational use. LPS acceptance testing will be performed by EDC. A Factory Acceptance Test (FAT) will be performed at GSFC by EDC personnel, prior to the shipment of the system to Sioux Falls, SD. This will ensure that the product leaving MOSDD is a quality product, and will help to isolate problems which may be encountered after shipment.

## 4.3 Prototyping

Prototyping will be utilized throughout the life-cycle, as deemed necessary, to clarify requirements, prove design feasibility, and resolve technical issues or mitigate key risk areas.

Prototyping will also be used in defining the operations interface to the LPS. This is a common use of prototyping. During the design phase, prototypes of the operations interface will be developed. These prototypes will be provided to the EDC operations personnel as a means of eliciting comments and determining additional requirements at the earliest possible juncture.

## 4.4 Computer-Aided System Engineering (CASE) Tools

The following section outlines the various Computer-Aided System Engineering (CASE) tools to be utilized in the development of the LPS.

### 4.4.1 Requirement Tracking

Requirements will be traced throughout the project life-cycle. The system level requirements will be traced to the Detailed Mission Requirements (DMR) provided by the Landsat 7 Project. Using the Requirements Traceability Mapping (RTM) tool which is now supported by Cadre, these requirements will be traced to each process specification during the software requirements phase, and down to the top module on each structure chart during the subsequent phases. The COTS package will aid the system engineers and developers in ensuring that all requirements are satisfied, and that the functionality maps back to the requirements set forth for the system. The RTM tool will be acquired by transferring the RTM tool license seat from the Pacor II project to the LPS.

### 4.4.2 Analysis and Design

The Cadre Teamwork tool has been used on previous MOSDD projects and will be utilized for the LPS. Utilizing the Site Management Agreement (SMA), Cadre seats no longer needed by other projects can be moved to the LPS free of charge. This will reduce the cost of COTS tools necessary for system development and provide the LPS with a knowledge base for the software development tools, in turn, providing a smaller learning curve. The Cadre Teamwork tool will be utilized to generate Data Flow Diagrams, Data Dictionary, Functional/Process Specifications, Structure Charts, Database Entity Relationship Diagrams (ERDs), and Module Specifications. The Cadre software will be hosted on a Sun Sparc 20 which has been procured for MOSDD.

### 4.4.3 Testing

Several current MOSDD projects utilize the Xrunner Package for testing. Additional information needs to be obtained about this COTS package.

### 4.4.4 Configuration Management

The LPS will use the Source Code Control Software (SCCS), which is available on all UNIX platforms for source code configuration management. This tool was preferred by the members of the development team who have utilized other

Configuration Management (CM) tools such as *PVCS Version Manager* which is used on several MOSDD projects.

In addition, MOSDD is currently developing a Configuration Change Request (CCR) automation tracking system for which the LPS will be the pilot project. The system is called the Information Processing Division (IPD) CCR Automation System (ICAS). ICAS has currently completed two deliveries and has scheduled a final delivery for the summer 1995. ICAS and LPS personnel have worked together to bring the LPS on-line during the spring of 1995.

### 4.4.5 Quality Assurance and Metrics

The LPS will utilize COTS metrics tools for measuring metrics as described in section 4.5.3. Current development efforts are utilizing a tool by Program Research Company called *PR-QA*. There is also a Software Process Assessment tool available from the 520 division which may be used as well. The Casevision Workshop Bundle tool described below advertises some complexity analysis as well. We will evaluate these tools to determine which best fits our needs.

### 4.4.6 Coding

Tool(s) such as the Casevision Workshop Bundle tool which is being procured for the LPS test bed machine will be utilized by the LPS. The Silicon Graphics, Inc. (SGI) Casevision tool provides a performance analyzer for software, test coverage tool, debugger, static analyzer, build analyzer, complexity analyzer, etc. This tool will be evaluated to ensure that it meets the needs of the project. If this tool does not meet the needs of the project, either because the tool itself does not meet our needs, or a different hardware architecture is selected, other tools will be evaluated.

## 4.5 Quality Assurance

The LPS will be looking at new and improved ways of developing quality software for reduced costs. LPS will be looking at various other projects to evaluate quality assurance techniques to be used on the LPS development effort.

### 4.5.1 Walkthroughs and Inspections

Walkthroughs and inspections will be conducted for design, PDL, and code beginning in the SRS phase. Representatives from the appropriate functions, other functions interfacing with the software, LPS system engineering, and the appropriate government personnel responsible for the functions will be expected to attend the software requirement and PDL walkthroughs. The code walkthroughs will be done by a much smaller group, perhaps only one other programmer. The key items to identify in the code walkthroughs will be inappropriate hard coding and approaches to coding that will hinder performance. Code will be available to civil servant as well SEAS personnel for inspection as deemed necessary. This will not be required for code certification, however.

The purpose of the walkthroughs is to reveal miscommunication or misunderstandings at the earliest possible time in the development life-cycle. In addition, the walkthroughs will verify design efficiency, complexity, and structure. Most importantly, the walkthroughs will be used as a mechanism to ensure that the requirements of the system are being met before system testing of the software.

Inspection certification forms are filled out for each item inspected at a walkthrough. SSDM provides these checklists; however, projects such as Spacelab have used modified versions of these checklists which have proven to be effective in gathering the metrics needed. The LPS will evaluate these inspection forms and modify them as necessary to meet the needs of the project. The project versions of the checklists will be provided in the software manager folder and be considered as additional information provided for the Software Management Plan.

### 4.5.2 Defect Casual Analysis (DCA)

Defect Casual Analysis (DCA) is a technique used to improve quality of software by preventing errors during software development.

The LPS will use this technique after the testing of each build. Problem reports will be categorized, causes of the problems will be discussed by developers, and recommendations for preventing the problems in the future will be presented. These recommendations will be submitted to management for action.

DCA is an activity typically done during the code development phase of a project. The LPS plans on conducting similar activities after each phase of the project to identify problems and recommendations to aid in the next phase of the project.

### 4.5.3 Metrics

Metrics will be maintained during the life-cycle to provide feedback to project personnel and for use by MOSDD. CASE tools will be used as necessary to aid in the gathering of the metrics. Examples of metrics to be captured are:

- Software sizing
- Total person hours spent on each life-cycle phase
- Code complexity
- Code structure quality
- Errors per DSI
- Errors per unit
- Rework effort for changes and error correction

### 4.5.4 Lessons Learned

At the end of each phase, lessons learned from the phase will be discussed and documented. At the end of the life-cycle, a comprehensive lessons learned document will be generated to aid in the development of future systems.

Typically, a lessons learned document is generated at the end of each project. However, many of the lessons learned from the early phases of a project are lost because the lessons were not gathered while there were still relatively fresh, and project personnel involved in the early phases of the project have since moved on to other projects and are not available for input to the lessons learned document. By taking a phased approach in the development of the lessons learned document, more information can be made available earlier for use by other projects.

### 4.5.5 Software Engineering Notebook (SEN)

The Software Engineering Notebook (SEN) is a workbook that consolidates the information pertinent to a software unit or module. There will be one SEN for each functional subsystem. SSDM states that these notebooks should contain the unit prologs, PDL, test plans, source code, etc. It will also be encouraged for the developers to expand these notebooks to include additional information. For example, information on how to execute the program using test tools, notes on why certain decisions or techniques were used, etc. These are notes that developers typically produce but do not keep in a consolidated area. This information is valuable information that is sometimes lost as time passes. The LPS will utilize these notebooks as an information source when it comes time to generate the as-built documentation or Programmer's Reference Manual.

The LPS software manager will also keep a notebook which will include the Software Management Plan, project standards, procedures, etc.

As with all documentation for the LPS, on-line documenting versus hard copy is the preferred method.

### 4.5.6 Quality Assurance Staffing

On past projects, a Product Assurance Officer (PAO) has been assigned nearly full time to a project. For the LPS, as a cost saving measure, the PAO will not be required to attend and review the PDL and code for certification walkthroughs.

The functions to be performed by the PAO include the collection and distribution of the metrics, performing the phase audits, and verification of the delivery package prior to system delivery to EDC.

## 4.6 Configuration Management (CM)

### 4.6.1 Functions

CM will be concerned with the development of procedures and standards for managing the development of the LPS.

CM will be responsible for identifying configuration items to be tracked (and tracking them), establishing and maintaining project baselines, and controlling change using the appropriate CCR mechanism. In addition, CM will be responsible for providing procedures and standards for the turnover and use of SCCS and other CM tools, as well as the building of the software executables, and preparing the delivery packages for EDC.

The CM group will adhere to the MOSDD CM Plan. The LPS project will utilize the MOSDD standard CCR form for requesting and tracking changes. All CCR requests and responses must be approved by the Project Configuration Management Board (PCMB). Changes to system level requirements must be approved by the MOSDD CCB.

Prior to a formal release of the system, ICCRs will be generated. These ICCRs will be tracked by CM. All ICCRs and analysis must be approved by the CRB which is chaired by the LPS software manager.

The CM personnel on the LPS will have a different role than has existed on past projects. With the introduction of the ICAS, the CM personnel will be responsible for the monitoring of CCR due dates, producing hard copy reports as needed by management, organizing the ICCR agenda for each CRB, organizing the CCR agenda for each PCMB, and working with the PAO to generate DCA metrics.

Due to LPS utilization of the ICAS, the CM personnel will be responsible for maintaining the correct build, release, and delivery dates within the ICAS. In addition, "canned" report scripts needed by the LPS project should be coordinated and created through the ICAS project management.

### 4.6.2 Staffing

Here again, the LPS is looking for some cost saving measures. With the implementation of the ICAS system, the responsibility of CM personnel in this area will be greatly reduced, thus reducing the need for the CM task. CM should be staffed accordingly.

## 4.7 Design and Implementation Guidelines

The following sections describe standards which will be followed in the design/implementation of the LPS software.

### 4.7.1 Documentation

The project will adhere to SSDM guidelines for data flow diagrams, data dictionary, structure charts, and document information.

The project will make an effort to minimize the amount of "printed" documentation that is necessary for the project. This should reduce project costs as well as reduce the documentation preparation lead time. On-line documentation will be maintained and baselined by appropriate personnel. Items, such as the data dictionary, may never need included in the paper version of a document, but will be considered part of the baseline of the appropriate documents and baselined accordingly. Once baselined, such on-line documents will continue to require CCRs to update and track changes.

It should be clearly noted that all work to be accomplished in each phase, as outlined by SSDM or this paper, is to be completed in the associated phase. All material that can be maintained and baselined on-line will be done as early as possible in its "native" application, i.e. Cadre, Microsoft Word, etc. If items are identified which cannot be maintained in this manner, hard copies will be provided. Directions for access to the on-line products, and, where possible, scripts to aid the reviewer in quickly printing the documentation, will be provided.

### 4.7.2 Operating System

The vendors being considered for the LPS operational system all provide Portable Operating System Interface (POSIX) compliant operating systems. Non-POSIX features of the system should be avoided; however, performance constraints may necessitate the use of these features. Any implementation which is not POSIX compliant should have approval by the LPS software manager, and should be isolated and clearly documented.

### 4.7.3 Unit Prologs and PDL

The project will adhere to the unit prolog format modified on a project basis as necessary by the LPS software manager. In addition, a standard PDL will be used.

### 4.7.4 Programming Languages

LPS software must conform to POSIX and American National Standards Institute (ANSI) standard C. All C code must pass successfully through the "lint" pre-compiler and checked for complexity, etc. utilizing a COTS package. Any deviation from the standard must be approved by the LPS software manager and clearly identified within the Prolog, PDL, and code of the identified unit. Every attempt should be made to isolate such code.

### 4.7.5 Database Management System (DBMS) and Operations Interface

Using past studies and project selections, the LPS will be utilizing the Oracle Database Management System (DBMS) unless cost prevent such a choice. Since the MOSDD has an abundance of experience with this database and has received excellent support from Oracle representatives, this should provide an excellent knowledge basis for the LPS.

Standard ANSI Structured Query Language (SQL) will be utilized where appropriate by the LPS. Any deviation from standard SQL must be approved by the LPS software manager and clearly identified the same method as mentioned above.

The forms generation packages which are provided with the chosen database will be used for as much of the user interaction with the system as possible. If functionality is identified, as necessary, which cannot be provided with the above mentioned package an appropriate GraphicalOperations Interface (GUI) package will be identified at that time. The GUI which best fulfills the needed functionality will be chosen.

### 4.7.6 Project Standards and Procedures

During the course of the project, many project procedures and guidelines will need to be developed and adhered to. Primary responsibility for the development of these procedures will be the software development group. These include procedures for, but not limited to:

- Cadre Teamwork model management
- Software requirements generation
- Structure chart standards

- Unit naming conventions
- PDL and prolog standards
- Certification process and forms to be followed
- COTS software usage procedures

### 4.7.7 Reuse

The LPS will identify a single person as their "reuse engineer". This person will be responsible for identifying areas of reuse within and outside of the LPS project. This includes reuse of design, algorithms, code, etc. As items for reuse are identified, they will be presented to the PCMB for project approval.

Since the LPS is being developed under code 500, the LPS will be required to look at the "renaissance" approach for areas of reuse and for providing building blocks to the renaissance libraries for reuse by future projects. A member of the LPS team, in conjunction with the Renaissance Group, will be responsible for reviewing the renaissance building blocks for reuse candidates within the LPS and making recommendations to the PCMB.

## 4.8 Training

Several types of training will be necessary during the course of the LPS project.

### 4.8.1 Software Development Personnel

Currently, the LPS project is looking at providing much of its training to the software development personnel through the use of SEAS personnel. This is a recommendation of LPS members who attended a Cadre training class. It was felt that the training was not adequate for our needs, therefore, not worth the associated cost.

As we are utilizing workstations for our development, it is imperative that software development personnel understand

and know information usually provided by persons with systems administration/programming experience. This is a change from development efforts in the past which have been done using mainframe technology. It is important for some members of the software development team to have knowledge of the kernel configurations, disk configurations and partitioning, daemons executing on the system, etc. in order to provide a workable system. No longer does the software development team just need to be concerned with the application software. By providing a system programmer/administrator person(s) on the development team who is also doing development, we will gain an area of expertise often missing on past projects. This expertise is key in the development of high performance systems such as the LPS. Training will be provided for these individuals when the system hardware is selected.

Training for all COTS products, such as Oracle, will be provided as sources and needs are identified.

### 4.8.2 System Test

Software development personnel will be required to provide training assistance to the system test personnel. Demonstrations of the system will be provided each build prior to turnover to the system test team. The Operations Guide will also be delivered with the software for each build in order that the system test team can verify the operations instructions contained within the document. This document, as all other documents for the LPS, may be contained on-line.

### 4.8.3 Acceptance Test/Operations

System testing will be required to train the appropriate acceptance test personnel for the EDC acceptance test teams. This acceptance team may include operations personnel. Demonstrations will be provided to the team members whenever feasible. System test personnel will support the acceptance test as needed by the EDC. A training plan between the LPS project and the EDC will be worked to ensure that all material needed or expected is identified between the two parties.

In support of training and the development effort, "hands on" versions of prototypes of the operations interface will be provided to EDC across the network. This will allow timely feedback into the implementation as well as provide early hands on training to the EDC personnel.

### 4.8.4 Maintenance

Training of the software maintenance group will be performed after the software has been transitioned to EDC.

## 4.9 Review Process

The LPS will be adopting a slightly different review process than other projects. Formal CCB reviews will not be given after each phase of the project, but at the conclusion of selected key phases throughout the project. For the other phases, informal project level reviews will given. These informal reviews will cover much more detail than a normal CCB review and may be staggered depending on the progress of the various groups. For instance if one group has completed preliminary design earlier than another, that group will give their informal review at that time so they may proceed on to detailed design. Any persons are invited to attend these informal reviews. It is expected that technical experts outside of the project will be able to attend these informal sessions. There are several benefits to this approach and some concerns.

Typically, work on the phases completes at some time prior to the scheduled review, then until the review is conducted, work does not progress at the same pace. This approach will ensure that progress is continually being achieved. In addition, informal reviews tend to stimulate much more technical inputs/questions than does the formal review process.

The concerns are that management will not have inputs at each phase, so when they do provide inputs, these inputs may require more effort to incorporate. Mitigation of this risk can be accomplished by providing slightly more technical detail at the monthly Project Status Reviews (PSRs). This will allow management to provide inputs on a monthly basis, actually an improvement over the current concept.

# 5 SEAS Contractor Evaluations

The software development activities for the LPS will be concentrated under a single task assignment along with the software integration and testing. Different from other task assignments in the past, each functional activity occurring under the LPS software task will be evaluated separately in the monthly evaluations. For example, each subsystem development area and the testing areas will be evaluated separately. This will provide a more accurate evaluation of the activities taking place under the task. Each evaluation will be approved and signed by the software manager who is serving as the ATR for the task assignment.

# 6 EDC Interfacing and Site Installation

## 6.1 System Development

Several items will need to be worked with the EDC team in support of the software development effort. EDC personnel will be invited to all reviews. Prototypes of the operations interface will be provided whenever possible for early feedback. Issues concerning transition will be documented in the Transition Plan.

The Transition Plan, to be generated by GSFC, will establish training requirements, training approach, roles and responsibilities for each group, problem reporting and resolution procedures, events and milestones etc.

## 6.2 System Installation

The software development team will support the installation of the LPS at the EDC in Sioux Falls, SD. The team will support the software installation and verification activities, as well as perform anomaly resolutions, as necessary.

## 6.2 System Operations

Throughout the development of the LPS, independent test/analysis tools will be created to verify and validate the LPS software. These test/analysis tools will be provided as unconfigured software, commonly known as "engineering" software, to the EDC operations and maintenance groups. In addition, since the LPS strings are designed to be completely independent, there will be, undoubtedly, tools to correlate output and database information across strings. These tools will also be provided to the EDC operations and maintenance groups.

# 7 Glossary

ANSI ----------American National Standards Institute

ATR ----------Assistant Technical Representative

BDR----------Build Design Review

BRR----------Build Readiness Review

CASE ----------Computer-Aided System Engineering

CCB----------Configuration Control Board

CCR----------Configuration Change Request

CM ----------Configuration Management

COTS----------Commercial Off-the-Shelf

CRB----------Configuration Review Board

DCA ----------Defect Casual Analysis

DDS----------Detailed Design Specification

DMR----------Detailed Mission Requirements

DSI ----------Delivered Source Instructions

EDC----------EROS Data Center

ERD----------Entity Relationship Diagram

EROS----------Earth Resources Observation System

ESMO----------Earth Science Mission Operations

ETM+ ----------Enhanced Thematic Mapper Plus

F&PS ----------Functional and Performance Specification

GSFC----------Goddard Space Flight Center

GTSIM----------Generic Telemetry Simulator

GUI----------Graphical User Interface

HWCI----------Hardware Configuration Item

ICAS----------IPD CCR Automation System

ICCR----------Internal Configuration Change Request

ICD ----------Interface Control Document

IDPS----------Image Data Processing Subsystem

IGS ----------International Ground Station

IPD----------Information Processing Division

LDTS ----------LPS Data Transfer Subsystem

LGS----------Landsat 7 Ground Station

LP DAAC--------Land Processes Distributed Active Archive Center

LPS----------Landsat 7 Processing System

MACS----------Management and Control Subsystem

Mbps----------Megabits per Second

MFPS----------Major Frame Processing Subsystem

MO&DSD---------Mission Operations and Data Systems Directorate

MOSDD ----------Mission Operations and Systems Development Division

NMOS----------Network and Mission Operations Support

NOAA ----------National Oceanic and Atmospheric Administration

Pacor II ---------Packet Processor II Data Capture Facility

PAO ----------Product Assurance Office

PDL----------Primitive Design Language

PCDS----------Payload Correction Data Processing Subsystem

PCMB----------Project Configuration Management Board

PMP----------Project Management Plan

POSIX ----------Portable Operating System Interface

PSR----------Project Status Review

RDCS----------Raw Data Capture Subsystem

RDPS ----------Raw Data Processing Subsystem

RTM----------Requirements Traceability Mapping

SCCS----------Source Code Control Software

SEAS ----------Systems, Engineering, and Analysis Support Services

SEN----------Software Engineering Notebook

SEWP----------Science and Engineering Workstation Procurement

SGI ----------Silicon Graphics, Inc.

SMA----------Site Management Agreement

SQL----------Structured Query Language

SRS----------Software Requirement Specification

SSDM----------SEAS System Development Methodology

SWCI ----------Software Configuration Item